

**Name: Maruya Razzak Riya**

**Id: 20-43371-1**

**Section: A**

## Activation functions

An activation function is a critical component of a neural network. It is used to determine the output of a neuron, or a set of neurons, given a set of inputs. Activation functions are also responsible for introducing non-linearity into the neural network, allowing for more complex decision boundaries and more accurate predictions. There are several types of activation functions, including Step, Sigmoid, ReLU, and Tanh. In this report, there will be a comparison of the characteristics of six of the most commonly used activation functions: Step, Sigmoid, Tanh, ReLU, ELU, and SELU. Each type has its own strengths and weaknesses, and the choice of activation function can significantly impact the performance of a neural network.

The step activation function is a binary function that returns either 0 or 1, depending on the input. It is the simplest of the activation functions and is most commonly used in binary classification problems. The main advantage of the step activation function is that it is computationally efficient and produces clear results. However, it can only be used in binary classification problems and is not suitable for more complex tasks. The main problem is that the gradient of the step function is zero. This makes the step function not so useful since, during back-propagation, it loses any knowledge of gradients in input values. A value that is 10000000 times the threshold has the same influence as the threshold value. Also, ignores any input value just below threshold. Additionally, it cannot be used for multi-class classification.

The Sigmoid function is a non-linear function that outputs values between 0 and 1. It is suitable for both classification and regression tasks, as it provides a smooth transition from 0 to 1. In multiclass classification, it returns the probability output for each class. Another advantage of this function is that, when used with  $(-\infty, +\infty)$  as in the linear function, it returns a value in the range of  $(0, 1)$ . This function has a few drawbacks, including the vanishing gradient problem. In case this is not a zero-centered function; a zero-centered function is one where the function range has 0 in the middle. Most machine learning algorithms work better with zero centers. Also, it is computationally expensive (exponential in nature).

The Tanh activation functions are also a type of continuous activation function similar to Sigmoid but are symmetric over the origin; also, the output values range between -1 and 1. This type of activation function is most commonly used in classification problems. Tanh will give zero-centered outputs. It also solves the problem of the values all being of the same sign. This function is non-linear in nature, so it can easily backpropagate the errors. There is a similar drawback to the sigmoid function; it still has the vanishing gradient problem. It is computationally expensive (exponential in nature).

ReLU (rectified linear unit) activation functions are a type of continuous activation function that returns either 0 or the input value depending on whether the input is greater than or less than 0. This type of activation function is most commonly used in deep learning networks. As the ReLU function is non-linear, it is simple to backpropagate errors and use it to activate neurons across many layers. Also, this function accelerates the convergence of stochastic gradient descent compared to the sigmoid and tanh activations. It is computationally efficient, allowing the network to converge very quickly. It is capable of outputting a true zero value. ReLU output is not zero-centered; it decreases the efficiency of the neural network. During backpropagation, the weights' gradients will either be uniformly positive or uniformly negative. There is a dying ReLU problem that occurs when the neuron gets stuck on the negative side and constantly outputs zero. This arises either when the learning rate is high or when the negative bias is significant.

The ELU (Exponential Linear Unit) function is similar to the ReLU function, but it outputs a negative value for negative inputs. As it ensures that neurons remain active even for negative inputs, ELU does not suffer from the problem of vanishing gradients and exploding gradients. Unlike ReLU, ELU does not suffer from the problems of dying neurons. Using ELU leads to a lower training time and higher accuracy in neural networks as compared to ReLU and its variants. The ELU activation function is continuous and differentiable at all points. Since ELUs can have negative values, it pushes the mean of the activations closer to zero. Having mean activations closer to zero also causes faster learning and convergence. ELU is slower to compute due to its non-linearity with negative inputs.

SELUs, or Scaled Exponential Linear Units, are activation functions that induce self-normalization. SELU network neuronal activations automatically converge to a zero mean and unit variance. This function is similar to the ELU function, but it is scaled to ensure that the output of the entire layer remains constant. This helps to reduce the problem of vanishing gradients, as it prevents the output from becoming too small. Compared to ReLUs, SELUs cannot die. SELUs learn faster and better than other activation functions without needing further processing. This is a relatively new activation function, so it is not yet used widely in practice.

Each activation function has its own strengths and weaknesses. The sigmoid function is simple to understand and implement but suffers from the vanishing gradient problem. The ReLU function is efficient and helps to alleviate the vanishing gradient problem but suffers from the problem of dead neurons. This problem is solved by ELU, but it has a slower test time. Existing problems are solved by SELU, but this function needs further research to be used more widely.

It is important to choose the appropriate activation function based on the specific task and dataset. Researchers and practitioners are still exploring and experimenting with different activation functions to improve the performance of deep learning models.